УДК 681.3.06

Л. Б. КАЩЕЕВ, канд. техн. наук, ***А. С. ЛИХУШИН***

ЗАДАЧА ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ ПРИ АНАЛИЗЕ РЫНОЧНЫХ КАРЗИН

В статті пропонується аналіз існуючих алгоритмів добутку знань в базах даних. На прикладі задачі аналізу корзин споживачів розглядаються методи обробки даних, показники підтримки достовірності та покращення добутку даних. Розглянуто механізм пошуку правил та існуючі алгоритми Apriori, DHP, DIC.

В последнее время неуклонно растет интерес к методам обнаружения знаний в базах данных. Весьма внушительные объемы современных баз данных вызвали устойчивый спрос на новые масштабируемые алгоритмы анализа данных. Для обнаружения скрытых знаний необходимо применять специальные методы, при помощи которых можно добыть знания из “завалов” информации. За этим направлением закрепился термин *добыча данных*.

Добыча данных – исследование и обнаружение алгоритмами скрытых знаний, которые ранее не были известны, нетривиальны, практически полезны, доступны для интерпретации человеком. Одной из наиболее распространенных задач анализа данных является определение часто встречающихся наборов объектов (анализ рыночных корзин) в большом множестве наборов. Для решения подобной задачи используются алгоритмы поиска ассоциативных правил.

Формальная постановка задачи формулируется следующим образом.

Объекты, составляющие исследуемые наборы данных, обозначаются множеством

$$\mathrm{I} = \{ i_1, i_2, \ldots, i_j, \ldots, i_n \}, \qquad (1)$$

где i_1 – объекты, входящие в анализируемые наборы; n – общее количество объектов.

Анализируемые наборы объектов из множества I, хранящиеся в базе данных, описываются как подмножество множества I:

$$T = \{ i_j \,|\, i_j \in \mathrm{I} \}. \qquad (2)$$

Такие подмножества применительно к потребительским корзинам соответствуют наборам товаров, покупаемых потребителем и сохраняемых в базах данных в виде товарного чека или накладной. Набор подмножеств (класс), информация о которых доступна для анализа, обозначаются следующим множеством:

$$D = \{ T_1, T_2, \ldots, T_r, \ldots, T_m \}, \qquad 3)$$

где m – количество доступных для анализа подмножеств.

Множество подмножеств, в которые входит объект i_j, обозначим следующим образом:

$$D_{i_j} = \left\{ T_r \middle| i_j \in T_r ; j = 1..n; r = 1..m \right\} \subseteq D . \quad (4)$$

Некоторый произвольный набор объектов обозначим следующим образом:

$$F = \left\{ i_j \middle| i_j \in I ; j = 1..n \right\} . \quad (5)$$

Набор, состоящий из k объектов, называется k-элементным набором.

Класс подмножеств, в которые входит набор F, обозначим следующим образом:

$$D_F = \left\{ T_r \middle| F \in T_r ; r = 1..m \right\} \subseteq D . \quad (6)$$

Отношение количества подмножеств, в которое входит набор F, к общему количеству подмножеств называется поддержкой (support) набора F и обозначается $Supp(F)$:

$$Supp(F) = \frac{|D_F|}{|D|} . \quad (7)$$

При поиске пользователь системы может указать минимальное значение поддержки интересующих его наборов $Supp_{min}$. Набор называется частым, если значение его поддержки больше минимального значения. Таким образом, при поиске ассоциативных правил требуется найти множество всех частых наборов (провести синквенциальный анализ):

$$L = \left\{ F \middle| Supp(F) > Supp_{min} \right\} . \quad (8)$$

Результаты, получаемые при решении задачи анализа потребительских корзин, принято представлять в виде правил вида *если (условие) то (результат)*, где *условие* – обычно не логическое выражение, а набор объектов из множества I, с которыми ассоциированы объекты, включенные в *результат* данного правила.

Основным достоинством ассоциативных правил является их легкое восприятие человеком и простая интерпретация языками программирования. Однако они не всегда полезны. Выделяют три вида правил:

- *Полезные правила* – содержат действительную информацию, которая ранее была неизвестна, но имеет логичное объяснение. Такие правила могут быть использованы для принятия решений, приносящих выгод;
- *Тривиальные правила* – содержат действительную и легко объяснимую информацию, которая уже известна;
- *Непонятные правила* – содержат информацию, которая не может быть объяснена, и требуют дополнительного анализа.

Для оценки полезности полученных правил вводятся следующие критерии.

Поддержка – показывает, какой процент транзакций поддерживает данное правило. Так как правило строится на основании набора, то, значит, правило $X \Rightarrow Y$ имеет поддержку, равную поддержке набора F, который составляют X и Y:

$$Supp_{X \Rightarrow Y} = Supp_F = \frac{|D_{F=X \cup Y}|}{|D|}. \quad (9)$$

Достоверность – показывает вероятность того, что из наличия в транзакции набора X следует наличие в ней набора Y. Достоверностью правила $X \Rightarrow Y$ является отношение числа подмножеств, содержащих наборы X и Y, к числу подмножеств, содержащих набор X:

$$Conf_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X|} = \frac{Supp_{X \cup Y}}{Supp_X}. \quad (10)$$

Очевидно, что чем больше достоверность, тем правило лучше. К сожалению, достоверность не позволяет оценить полезность правила. Если процент наличия в транзакциях набора Y при условии наличия в них набора X меньше, чем процент безусловного наличия набора Y, т.е.:

$$Conf_{X \Rightarrow Y} = \frac{Supp_{X \cup Y}}{Supp_X} < Supp_Y, \quad (11)$$

Это значит, что вероятность случайно угадать наличие в подмножестве набора Y больше, чем предсказать это с помощью правила $X \Rightarrow Y$. Для исправления такой ситуации вводится мера – *улучшение*.

Улучшение – показывает, полезнее ли правило случайного угадывания. Улучшение правила является отношением числа транзакций, содержащих наборы X и Y, к произведению количества транзакций, содержащих набор X, и количества транзакций, содержащих набор Y:

$$impr_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X||D_Y|} = \frac{Supp_{X \cup Y}}{Supp_X \cdot Supp_Y}. \quad (12)$$

Если улучшение больше единицы, то это значит, что с помощью правила предсказать наличие набора Y вероятнее, чем случайное угадывание, если меньше единицы, то наоборот.

Значения для параметров минимальная поддержка и минимальная достоверность выбираются таким образом, чтобы ограничить количество найденных правил. Если поддержка имеет большое значение, то алгоритмы будут находить правила, хорошо известные аналитикам или настолько очевидные, что нет никакого смысла проводить такой анализ. С другой стороны, низкое значение поддержки ведет к генерации огромного количества правил, что, конечно, требует существенных вычислительных ресурсов. Тем не менее, большинство интересных правил находится именно

при низком значении порога поддержки. Хотя слишком низкое значение поддержки ведет к генерации статистически необоснованных правил.

Выявление частых наборов объектов – задача требующая большое количество вычислений, а следовательно, и времени. Один из первых алгоритмов, эффективно решающих подобный класс задач, – это алгоритм Apriori, разработанный в 1994 году Рамакришнаном и Агравалом. Кроме этого алгоритма в последнее время был разработан ряд других алгоритмов: DHP, Partition, DIC [1,2], которые используются также для решения других задач поиска ассоциативны правил(обобщенные ассоциативные правила, численные ассоциативные правила). Алгоритм Apriori превосходит другие по простоте реализации и скорости выполнения и более применим к данной задаче. Следует упомянуть и алгоритм AprioriTid, являющийся развитием алгоритма Apriori.

Алгоритм использует одно из свойств поддержки, гласящее: поддержка любого набора объектов не может превышать минимальной поддержки любого из его подмножеств:

$$Supp_F \le Supp_E, при \quad E \subset F . \tag{13}$$

Алгоритм Apriori определяет часто встречающиеся i-элементные наборы. Каждый этап состоит из двух шагов: формирования кандидатов и подсчета поддержки кандидатов.

На i-ом этапе на шаге формирования кандидатов алгоритм создает множество кандидатов из i-элементных наборов, чья поддержка пока не вычисляется. Для подсчета поддержки кандидатов нужно сравнить каждую транзакцию с каждым кандидатом. Очевидно, что количество кандидатов может быть очень большим, поэтому гораздо эффективнее использовать подход, основанный на хранении кандидатов в хеш-дереве.

После того как найдены все часто встречающиеся наборы элементов, можно приступить непосредственно к генерации правил.

Извлечение правил – менее трудоемкая задача. Во-первых, для подсчета достоверности правила достаточно знать поддержку самого набора и множества, лежащего в условии правила. Например, имеется часто встречающийся набор $\{A,B,C\}$ и требуется подсчитать достоверность для правила $AB \Rightarrow C$. Поддержка самого набора нам известна, но и его множество $\{A,B\}$, лежащее в условии правила, также является часто встречающимся в силу свойства антимонотонности, и значит его поддержка известна. Тогда мы легко сможем подсчитать достоверность. Это избавляет от нежелательного просмотра базы транзакций, который потребовался в том случае, если бы это поддержка была неизвестна.

Чтобы извлечь правило из часто встречающегося набора *F*, следует найти все его непустые подмножества. И для каждого подмножества s мы сможем

сформулировать правило $s \Rightarrow (F - s)$, если достоверность правила $Conf(s \Rightarrow (F - s)) = \dfrac{Supp(F)}{Supp(s)}$ не меньше порога $minconf$.

Заметим, что числитель остается постоянным. Тогда достоверность имеет минимальное значение, если знаменатель имеет максимальное значение, а это происходит в том случае, когда в условии правила имеется набор, состоящий из одного элемента. Все супермножества данного множества имеют меньшую или равную поддержку и, соответственно, большее значение достоверности. Это свойство может быть использовано при извлечении правил.

Если мы начнем извлекать правила, рассматривая сначала только один элемент в условии правила, и это правило имеет необходимую поддержку, тогда все правила, где в условии стоят супермножества этого элемента, также имеют значение достоверности выше заданного порога. Например, если правило $A \Rightarrow BCDE$ удовлетворяет минимальному порогу достоверности $minconf$, тогда $AB \Rightarrow CDE$ также удовлетворяет. Для того чтобы извлечь все правила используется рекурсивная процедура. Важное замечание: любое правило, составленное из часто встречающегося набора, должно содержать все элементы набора. Например, если набор состоит из элементов $\{A,B,C\}$, то правило $A \Rightarrow B$ не должно рассматриваться.

Если объекты имеют дополнительные атрибуты, которые влияют на состав объектов в подмножествах, а следовательно, и в наборах корзин, то они должны учитываться в генерируемых правилах. В этом случае условия части правил будут содержать не только проверку наличия объекта в подмножествах, но и более сложные операции сравнения: больше, меньше, включение, симметрическая разность и др.

Задача поиска ассоциативных правил впервые была представлена для анализа рыночной корзины. Ассоциативные правила эффективно используются в сегментации покупателей по поведению при покупках, анализе предпочтений клиентов, планировании расположения товаров в супермаркетах, кросс-маркетинге, адресной рассылке. Однако сфера применения этих алгоритмов не ограничивается лишь одной торговлей. Их также успешно применяют и в других областях: медицине, для анализа посещений веб-страниц, для анализа текста, для анализа данных по переписи населения, в анализе и прогнозировании сбоев телекоммуникационного оборудования и т.д.

Список литературы: 1. *Барсегян А.А., Куприянов М.С. и др.* Методы и модели анализа данных: OLAP и Data Mining. СПб.: БХВ-Петербург, 2004. – 336 с. **2.** *Куприянов М.С., Ярыгин О.Н.* Построение отношения и меры сходства нечетких объектов//Техническая кибернетика, 1988, №3. С.78 – 86.

Поступила в редколлегию 03.05.06